Jurnal Algoritma, Logika dan Komputasi
Vol.I (No. 1) : 36 - 43. Th. 2018
ISSN: 2620-620X

WATERMARK DENGAN GABUNGAN STEGANOGRAFI DAN *VISIBLE WATERMARKING*

Watermark Using Steganography and Visible Watermarking

Juni Rosmiyati, jrosmiyati@gmail.com[1)], Teady Matius Surya Mulyana, tmulyana@bundamulia.ac.id[2)]
[1)2)]Teknik Informatika / Fakultas Teknik Dan Desain, Universitas Bunda Mulia

ABSTRACT

Images, that are currently circulating on the internet are many, such as advertising images for the promotion of products from a particular brand / institution, the image of the digital image, photographs of the works of famous photographers and many other examples. To identify each image appropriately, it can be added with a special logo or sign that is easily recognizable to the user. But the current advanced technology also supports duplication of the same logo / sign by irresponsible parties. Using the LSB and Grayscale methods, information such as text can be inserted into the watermark image that you want to use. Where there is a combination between the steganography process and the watermark process in one medium so that it can contain information visually and invisible. By using a stegowater application that incorporates stenographic and watermark processes, the user can use it to identify the authenticity of the image being processed, has been through the modification process or not.
Keywords: *Image, Steganografi, Watermark*

ABSTRAK

Citra, gambar atau *image* yang saat ini beredar di internet sangatlah banyak, misal gambar iklan untuk promosi produk dari suatu *brand*/institusi tertentu, citra hasil karya *digital image*, foto pemandangan hasil karya fotografer terkenal dan masih banyak contoh lain. Untuk mengidentifikasi masing-masing citra dengan tepat, dapat ditambahkan dengan logo atau tanda khusus yang mudah dikenali oleh *user*. Tetapi teknologi yang maju saat ini juga mendukung dapat dilakukannya duplikasi dari logo/ tanda yang sama oleh pihak yang tidak bertanggung jawab. Dengan menggunakan metode LSB dan Grayscale, informasi seperti teks dapat disisipkan ke dalam citra *watermark* yang ingin digunakan. Dimana terjadi penggabungan antara proses *steganografi* dan proses *watermark* dalam satu media sehingga dapat berisi informasi secara *visible* dan *invisible*. Dengan menggunakan aplikasi *stegowater* yang menggabungkan proses *steganografi* dan *watermark*, *user* dapat menggunakannya untuk mengidentifikasi keaslian dari citra yang diproses, telah melalui proses modifikasi atau tidak.
Kata Kunci: Citra, *Steganographi*, *Watermark*

PENDAHULUAN

Istilah *stegowater* berasal dari dua kata yang digabungkan, yaitu *steganografi* dan *watermark.*

Dimana kata "*steganografi*" berasal dari bahasa Yunani *steganos*, yang artinya "tersembunyi atau terselubung", dan *graphein*, "menulis". Sehingga *steganografi* adalah seni dan ilmu menulis pesan tersembunyi atau menyembunyikan pesan dengan suatu cara. *Steganografi* membutuhkan dua properti, yaitu wadah penampung dan data rahasia yang akan disembunyikan. *Steganografi* digital merupakan media digital sebagai wadah penampung, misalnya citra, *audio*, teks, dan *video*. Data rahasia yang disembunyikan juga dapat berupa citra, *audio*, teks atau *video*.

Sedangkan untuk *watermark,* Guo (Guo, 2009) menjelaskan 2 jenis *watermark*, yaitu *visible watermarking* dan *invisible watermarking*. Braudaway (Braudaway, 1996), Men (Meng, 1998), Kankanhalli (Kankanhalli, 1999) dan Chen (Chen, 2000) seperti yang dikutip oleh Guo

Jurnal Algoritma, Logika dan Komputasi
Vol.I (No. 1) : 36 - 43. Th. 2018
ISSN: 2620-620X

(Guo, 2009) menjelaskan bahwa *visible watermarking* merupakan suatu proses pembentukan citra ter*watermark* dimana citra *watermark* ditampilkan pada citra ter*watermark* secara kasat mata. Teknik *watermarking* bekerja dengan menyisipkan sedikit informasi yang menunjukkan kepemilikan, tujuan, atau data lain pada media digital tanpa mempengaruhi kualitasnya.

METODE PENELITIAN

Dalam masing-masing proses, digunakan metode tertentu untuk mendukung proses yang berjalan, seperti dalam *steganografi* menggunakan metode LSB dan proses *watermark* menggunakan metode *grayscale* dan LSB.

Least Significant Bit (LSB) adalah algoritma yang dalam prosesnya membaca bit dari sumber tertentu sehingga dapat dilakukan perubahan nilai bit terbelakang sesuai dengan bit yang diinginkan. Dalam proses stegowater, metode LSB dilibatkan pada proses *steganografi*, pembuatan *header*, dan setelah proses *watermark.*

Tahapan dalam implementasi LSB terbagi menjadi 3 yaitu:

Tahap Pertama: Memperoleh nilai bit, baik dari sisi citra maupun dari *stego-teks* yang akan digunakan sebagai nilai bit terbelakang. Citra memperoleh nilai bitnya dari intensitas citra, sedangkan stego-teks dapat memanfaatkan kode ASCII dalam konversi huruf ke nilai biner atau sebaliknya.

Tahap Kedua: Nilai desimal yang diperoleh diolah Untuk mendapatkan nilai bit terkecil dapat memanfaatkan persamaan (1).

$$k = B_{st} \bmod 2 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

dimana:
k : Nilai *bit* terkecil dari Bst
B_{st} : Biner dari citra/*stego-teks*

Tahap Ketiga: Memasukkan nilai bit ke dalam nilai bit citra.

Jika Bc(x,y) di mod dengan 2 dan bernilai k, maka tidak terjadi perubahan nilai pada Bc(x,y). Tetapi jika tidak, maka akan diuji nilai k. Apabila k bernilai 0, maka dilanjutkan ke persamaan (2), sedangkan jika k bernilai 1, maka dilanjutkan ke persamaan (3).

$$Bc'(x,y) = Bc\,(x,y) - 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$
$$Bc'(x,y) = Bc\,(x,y) + 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

dimana:
Bc(x,y) : Nilai *bit* citra
Bc'(x,y): Nilai *bit* citra setelah di proses
k : Nilai *bit* terkecil dari Bst

Mulyana (Mulyana, 2013) menjelaskan bahwa, jika suatu citra watermark dijadikan citra *grayscale*, yang hanya mempunyai satu nilai intensitas, maka jika nilai intensitas pada piksel-piksel dari citra watermark yang telah diubah menjadi citra *grayscale* tersebut digunakan untuk menaikkan atau menurunkan nilai intensitas channel RGB suatu citra dengan posisi dan variasi intensitas masing-masing piksel yang sama dengan citra *watermark* akan menghasilkan variasi gelap dan terang pada piksel-piksel citra yang akan diberi *watermark* dengan posisi-posisi yang sama dengan posisi pada citra *watermark*. Di dalam proses *stegowater*, metode *Grayscale* hanya dilibatkan dalam proses *watermarking* saja.

Menurut Mulyana (Mulyana, 2013), proses *watermarking* dengan metode *grayscale* melalui lima tahapan, yaitu:

Tahap Pertama*: Preprocessing*, penyiapan citra *watermark*, dimana citra *watermark* akan diproses pengubahan menjadi abu yang dapat dilakukan dengan berbagai cara, salah satunya dapat dilakukan dengan metode sederhana seperti dikutip oleh Simarmata (Simarmata, 2007) dan Sutoyo (Sutoyo,2009), dapat dilihat pada persamaan (4).

$$f(x,y) = (R(x,y) + G(x,y) + B(x,y)) / 3 \quad \ldots\ldots\ldots (4)$$

dimana:
f(x,y) : pixel abu-abu
R(x,y) : intensitas channel Red
G(x,y) : intensitas channel Green

Jurnal Algoritma, Logika dan Komputasi
Vol.I (No. 1) : 36 - 43. Th. 2018
ISSN: 2620-620X

B(x,y) : intensitas channel blue

Tahap Kedua: *Bitmaping*, pemindahan citra *watermark* yang sudah menjadi citra abu-abu akan dipindahkan ke *bitmap*, bersamaan dengan proses ini piksel yang dianggap latar belakang diset dengan nilai 0, sedangkan piksel latar depan akan diset nilai apa adanya. Pada tahap ini, ditentukan nilai intensitas terkecil dari latar depan citra.

Tahap Ketiga: Pembuatan bitmap penapis, penggantian nilai piksel latar depan citra *watermark* pada *bitmap* dengan nilai patokan yang didapat dari nilai terendah intensitas citra *watermark* (atau dengan nilai lainnya) dengan konstanta. Proses hanya dilakukan pada piksel yang nilai intensitasnya bukan 0, atau piksel latar belakang. Nilai pengganti didapat dengan persamaan (5).

$$G'(x,y) = \min(G(x,y)) + G(x,y) + K \quad \text{.....(5)}$$

dimana:
G'(x,y) : Bitmap Citra Watermark
G(x,y) : Citra Watermark
K : Konstanta

Tahap Keempat: Operasi penjumlahan pada citra utama dengan nilai pada bitmap *watermark*. Pada operasi ini setiap piksel pada bitmap penapis sudah mengandung suatu nilai penapis, karena itu operasi penjumlahan cukup dengan menjumlah piksel-piksel pada citra utama dengan piksel-piksel pada bitmap citra *watermark* tanpa mengalikan dengan nilai penapisnya. Persamaan penjumlahan itu dapat dilakukan dengan persamaan (6).

$$F(x,y) = C(x,y) + G'(x,y) \quad \text{.....(6)}$$

dimana:
F(x,y) : Citra hasil *watermarking*
C(x,y) : Citra utama
G'(x,y) : *Bitmap* Citra *Watermark*

Tahap Kelima: Setelah nilai pada F(x,y) berubah sesuai tahapan yang sudah ditetapkan, nilai *bit* terakhir dari F(x,y) harus dikembalikan dalam bentuk yang sama dengan menggunakan persamaan (7).

Setelah proses *watermark* selesai, dilakukan proses untuk menghitung jumlah karakter *stego-teks* maksimal yang dapat diinput ke dalam citra *watermark* dengan menggunakan persamaan (7) berikut.

$$J_{char} = ((W_{width} - S_x) \text{ div } 8) * ((W_{height} - S_y) \text{ div } 8) * 8 - 2 \quad \text{.....(7)}$$

dimana:
S_x : Posisi x dari *stego-teks*
S_y : Posisi y dari *stego-teks*
W_{width} : Lebar dari citra *watermark*
W_{height} : Tinggi dari citra *watermark*
J_{char} : Jumlah *stego-teks*

Header merupakan bagian informasi data yang disimpan ke dalam citra yang dituju, dengan tujuan untuk mempermudah proses *Read*.

Berdasarkan tempat penyimpanan, header terbagi menjadi 2, yaitu:
1. *Header* S, yang terletak di dalam citra *watermark*, berisi informasi tentang letak dari stego-teks dan besar ukuran stego-teks. Contoh : |10,10;181;15|
2. *Header* W, yang terletak di citra dasar, berisi informasi dari letak citra watermark, arah penyimpanan dan besar ukuran citra *watermark*. Contoh : |10,10;181h;15|

Berdasarkan arah penyimpanan, *header* juga terbagi dalam 2, yaitu:
1. Horizontal (h), dimana data disimpan secara horizontal (kiri ke kanan), yang terpengaruh oleh posisi x yang dismasukkan dan lebar (*width*) dari citra. Contoh : |10,10;181h;15|
2. Vertikal (v), dimana data tersimpan secara vertikal (atas ke bawah), yang terpengaruh oleh posisi y yang dimasukkan dan tinggi (*height*) dari citra. Contoh : |10,10;180v;15|

Dalam pembuatan *header* S, tahapan yang dilalui terdiri dari 4 tahap, yaitu:

Tahap Pertama: Membuat *header* sesuai dengan format yang sudah ditentukan, dimana *height* dan *weight*

Jurnal Algoritma, Logika dan Komputasi
Vol.I (No. 1) : 36 - 43. Th. 2018
ISSN: 2620-620X

menentukan persamaan yang digunakan. *Width* menggunakan persamaan (8) dan (9), sedangkan *height* menggunakan persamaan (10) dan (11).

$H_s = |W_x + S_x, W_y + S_y; W_x + W_{width}; W_y + Y_{max}|$.....(8)

$Y_{max} = ((W_{height} - S_y) \text{ div } 8) * 8 + S_y$ .....(9)

$H_s = |W_x + S_x, W_y + S_y; W_y + W_{height}; W_x + X_{max}|$.....(10)

$X_{max} = ((W_{width} - S_x) \text{ div } 8) * 8 + S_x$ .....(11)

dimana:

S_x : Posisi x dari *stego-teks*
S_y : Posisi y dari *stego-teks*
W_x : Posisi x dari *watermark*
W_y : Posisi y dari *watermark*
X_{max} : Batas X dari citra *watermark*
Y_{max} : Batas Y dari citra *watermark*
W_{width}: Lebar citra *watermark*
W_{height}: Tinggi citra *watermark*

Tahap Kedua: Mengkonversi nilai *string* dari *header* ke dalam kode ASCII. Contoh : |10,10;181;15| = 129 49 48 44 49 48 59 49 56 49 59 49 53 124

Tahap Ketiga: Mengkonversi kode ASCII dari *header* ke dalam bentuk biner.

124 : 01111100
49 : 00110001
48 : 00110000
44 : 00101100
59 : 00111011
56 : 00111000
53 : 00110101

Tahap Keempat: Memasukkan bilangan biner yang diperoleh ke dalam citra *watermark* dengan menggunakan metode LSB.

Sedangkan dalam pembuatan *header* W, tahapan yang dilalui terdiri dari 4 tahap, yaitu:

Tahap Pertama: Membuat *header* sesuai dengan format yang sudah ditentukan, dimana *height* dan *weight* menentukan persamaan yang digunakan. *Width* menggunakan persamaan (12), sedangkan *height* menggunakan persamaan (13).

$H_w = |W_x,W_y;W_x+W_{width}+dir;W_y+4|$.....(12)

$H_w = |W_x,W_y;W_x+W_{height}+dir;W_x+4|$ .....(13)

dimana:

W_x : Posisi x dari *watermark*
W_y : Posisi y dari *watermark*
W_{width} : Lebar citra *watermark*
W_{height} : Tinggi citra *watermark*
dir : Arah penyimpanan *header*

Tahap Kedua: Mengkonversi nilai *string* dari *header* ke dalam kode ASCII. Contoh : |10,10;181h;15| = 129 49 48 44 49 48 59 49 56 49 104 59 49 53 124

Tahap Ketiga: Mengkonversi kode ASCII dari header ke dalam bentuk biner.

124 : 01111100
49 : 00110001
48 : 00110000
44 : 00101100
59 : 00111011
56 : 00111000
104 : 01101000
53 : 00110101

Tahap Keempat: Memasukkan bilangan biner yang diperoleh ke dalam citra dasar dengan menggunakan metode LSB.

Metode LSB digunakan juga dalam proses *Read*, untuk membaca nilai *bit* yang tersimpan melalui lima tahapan, yaitu:

Tahap Pertama: Nilai piksel yang diperoleh dari citra *stegowater* diubah ke dalam bentuk biner.

254 : 1111 1110
255 : 1111 1111

Tahap Kedua: Nilai biner yang diperoleh, dibaca hanya *bit* terakhirnya saja menggunakan persamaan (1).

1111 0000 : 0
1111 1111 : 1

Tahap Ketiga: Bit disusun secara beruturan sesuai dengan letaknya, lalu dibagi per 8 bit seperti yang ditunjukkan dalam Gambar 1.

Jurnal Algoritma, Logika dan Komputasi
Vol.I (No. 1) : 36 - 43. Th. 2018
ISSN: 2620-620X

0	1	1	1	1	1	0	0
0	1	1	0	1	0	1	0
0	1	1	1	0	1	0	1
0	1	1	0	1	1	1	0
0	1	1	0	1	0	0	1
0	0	1	0	1	1	1	0
0	1	1	1	0	1	1	0
0	1	1	1	1	0	0	0

Gambar 1. Bit Terakhir yang Disusun

Tahap Ketiga: Dilakukan konversi nilai biner yang didapat, tiap 8 *bit* biner ke dalam bilangan desimal.

01111100 : 124
01101010 : 106
01110101 : 117
01101110 : 110
01101001 : 105
00101110 : 46
01110010 : 114
01111100 : 124

Tahap Kelima: Bilangan desimal yang diperoleh, diubah ke dalam kode ASCII.
124 106 117 110 105 46 114 124 : |juni.r|

HASIL DAN PEMBAHASAN

Keuntungan yang diperoleh dengan menggunakan metode LSB antara lain:

1. Perubahan yang terjadi tidak mudah dideteksi secara kasat mata.
2. Mutu dari citra penampung tidak jauh berubah dibanding aslinya.
3. Ukuran dari data yang bisa disimpan tergantung dari ukuran citra penampung, dimana makin besar ukuran citra penampung, makin banyak data (untuk sementara berupa teks) yang bisa disimpan di dalamnya.

Sedangkan kekurangan dari metode LSB adalah tidak tahan terhadap manipulasi citra seperti *crop, rotate, hue, saturation*, dan lain-lain yang dapat mengakibatkan berubahnya nilai *bit* ataupun posisi datanya. Dimana kekurangan ini dapat digunakan untuk mengetahui apakah citra tersebut telah mengalami proses modifikasi atau tidak.

Keuntungan dari penggunaan metode *Grayscale* adalah dapat mempermudah proses *watermark* sehingga menjadi lebih cepat dibandingkan dengan menggunakan ketiga *channel* warna sekaligus (RGB).

Sedangkan kelemahannya pada metode *Grayscale*, yaitu:

1. Apabila nilai dari latarnya atau citra dasar (*base*) mencapai 255, maka tidak dapat terjadi proses watermarking ke dalam citra tersebut. Sehingga diperlukan suatu metode yang dapat digunakan untuk mengatur tinggi rendahnya nilai tunggal *Grayscale*.
2. Jika terjadi *thresholding*, maka *watermark* pada piksel yang di *threshold* tersebut tidak terlihat jelas.

Percobaan berikut diuji pada *software Stegowater* v1.4.02.

Gambar 2. Citra dasar (*base*)

Gambar 3. Citra watermark

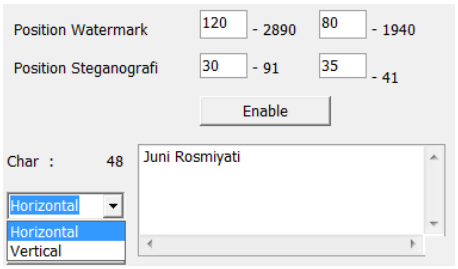
Proses dalam *stegowater* terbagi menjadi 2, yaitu *read* (membaca) dan *write* (menulis). Proses *write* terbagi menjadi empat bagian, yaitu:

1. Input citra dasar (*base*)
2. Input citra watermark

Jurnal Algoritma, Logika dan Komputasi
Vol.I (No. 1) : 36 - 43. Th. 2018
ISSN: 2620-620X

3. Input nilai posisi steganografi, watermark dan stego-teks
4. Output hasil citra *stegowater*

Citra pada Gambar 3 d*i-watermark* ke dalam Gambar 2, lalu diatur nilai posisi *watermark* dan *steganograf*, serta isi dari *stego-teks* seperti pada Gambar 4. Hasil akhir dari proses adalah citra *stegowater* seperti yang ditunjukkan dalam Gambar 5.

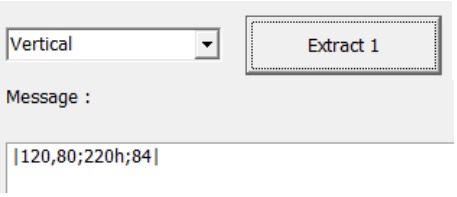


Gambar 4. Nilai posisi *watermark*, steganografi dan isi stego-teks.

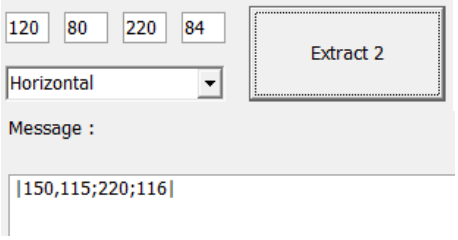
Gambar 5. Citra *stegowater* yang berisikan citra *base*, citra *watermark* dan stego-teks di dalamnya.

Citra *stegowater* seperti pada Gambar 5, yang diperoleh dari proses *write* memiliki data *stego-teks* didalamnya. Dimana isi stego-teks tersebut dapat dibaca kembali dengan menggunakan proses *Read* dalam aplikasi *stegowater*. Proses *Read* terbagi menjadi 2 tahap, yaitu:

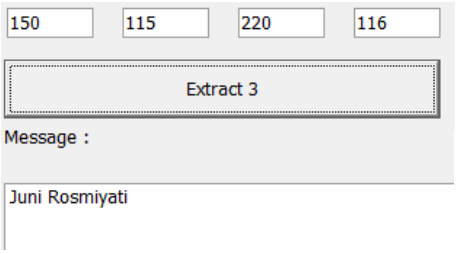
1. *Input* citra *stegowater*
2. *Input* dan *output* proses ekstrak



Gambar 6. Proses Ekstrak 1



Gambar 7. Proses Ekstrak 2



Gambar 8. Proses Ekstrak 3

Setelah citra stegowater dipilih oleh *user*, proses pada Gambar 7 dan selanjutnya dapat dilanjutkan. Gambar 7 menunjukkan kode *header* untuk *watermark*, sedangkan pada Gambar 6 menunjukkan kode *header* untuk *steganografi*. Dan pada Gambar 8, menunjukkan lokasi dari *stego-teks* dan isi *stego-teks*.

Dilakukan uji proses modifikasi pada citra *stegowater* yang dihasilkan, seperti pengubahan *brightness, contrast, crop*, rotasi, *flip*, saturasi, *hue*, *luminesce* dan *zoom* untuk melihat apakah *stego-teks* dan *header* masih lengkap setelah mengalami proses modifikasi. Hasil ujicoba dapat dilihat pada Tabel 1.

dimana:
+ : Header dari ekstrak benar
! : Terdapat bagian yang benar tetapi tidak sempurna
- : Hasil muncul tapi salah semua
o : Hasil tidak muncul sama sekali

Pada citra *stegowater* yang dilakukan perubahan *brightness* sebesar +45 dari awalnya, hasil yang diperoleh pada, Ekstrak 1 tetap menghasilkan *header* yang semestinya, tetapi pada ekstrak 2, *header* yang muncul terdapat bagian benar

Jurnal Algoritma, Logika dan Komputasi
Vol.I (No. 1) : 36 - 43. Th. 2018
ISSN: 2620-620X

tetapi tidak sempurna, dan terakhir pada ekstrak 3, isi stego-teks tidak muncul sama sekali.

Tabel 1. Hasil Uji Perlakuan

No	Perlakuan	Ekstrak 1	Ekstrak 2	Ekstrak 3
1	Brightness +45	+	!	o
2	Brightness +59	o	-	-
3	contrast +30	o	o	o
4	crop	o	o	o
5	Flip Horisontal	o	o	o
6	Flip Vertikal	o	o	o
7	Luminescen +50	o	o	o
8	Luminescen -80	o	o	o
9	Rotasi 180°	o	o	o
10	Zoom	o	o	o
11	Saturation -100	+	+	+
12	Saturation -67	+	+	+
13	hue +50	+	+	+
14	hue -50	+	+	+
15	hue -100	+	+	+

Perubahan *brightness* sebesar +59 menghasilkan nilai ekstrak 1, *header* W dari percobaan 8 masih benar. Dan hasil yang diperoleh, ekstrak 1 ternyata tidak menghasilkan *header*, dan perbedaan juga terdapat pada ekstrak 2 dan ekstrak 3 yang memunculkan hasil, tetapi salah semua.

Pada beberapa perlakuan berikut ini:

- Pengubahan *contrast* sebesar +30.
- Pemotongan ukuran citra / *crop* hingga berukuran 1560x1100px.
- Pembalikkan citra/ *flip* secara horizontal.
- Pembalikkan citra/ *flip* secara vertikal.
- Luminesce citra *stegowater* sebesar +50.
- Luminesce citra *stegowater* sebesar -80.
- Rotasi pada citra *stegowater* dengan rotasi sebesar 180 derajat, *clockwise*.
- Perbesaran pada citra *stegowater* hingga mencapai ukuran 1600x1200px.

Hasil ekstrak 1, 2 dan 3 dari percobaan nomor 3 sampai 10 tidak menghasilkan apapun. Sehingga modifikasi tersebut adalah jenis modifikasi yang dapat merusak nilai *header* dan stego-teks yang tersimpan dalam citra *stegowater.*

Pada percobaan nomor 11 sampai 15 citra *stegowater* dilakukan

- perubahan saturasi citra *stegowater* sebesar -100.
- Perubahan saturasi citra *stegowater* sebesar -67.
- Perubahan nilai *hue* dari citra *stegowater* sebesar +50.
- Perubahan nilai *hue* dari citra *stegowater* sebesar -50.
- Perubahan nilai *hue* sebesar -100 pada citra *stegowater.*

Hasil percobaan tersebut memberikan hasil positif pada ketiga jenis ekstrak, dimana ekstrak 1 berhasil membaca *header* W, ekstrak 2 berhasil membaca *header* S dan ekstrak 3 dapat mengetahui isi dari stego-teks secara sempurna. Sehingga jenis perlakuan pada percobaan 11 sampai 15 adalah jenis perlakuan yang tidak merusak nilai *header* dan stego-teks yang tersimpan dalam citra *stegowater*, meskipun perubahan warna pada citra *stegowater* lebih signifikan dibanding proses modifikasi lainnya.

SIMPULAN

Berdasarkan hasil uji modifikasi pada citra, proses modifikasi seperti *brightness, contrast, rotate,* dan *zoom* dapat merusak *header* dan *stego-teks* yang tersimpan di dalam citra *stegowater*. Sedangkan proses modifikasi seperti *hue* dan *saturation* tidak menyebabkan hilangnya *stego-teks* ataupun *header* di dalam citra *stegowater.*

Saran

1. Berdasarkan hasil pengujian yang diperoleh, penggunaan stego-teks di dalam citra dapat diaplikasikan pada logo sertifikasi digital sehingga dapat digunakan untuk mengetahui valid tidaknya suatu sertifikasi secara digital.
2. Dapat menambahkan unsur *kriptografi* unuk meningkatkan keamanan dari aplikasi seperti *Caesar Cipher* dalam konversi kode ASCII ke dalam biner

Jurnal Algoritma, Logika dan Komputasi
Vol.I (No. 1) : 36 - 43. Th. 2018
ISSN: 2620-620X

atau sebaliknya sehingga lebih *secure* dalam melindungi data yang ada serta sekaligus melengkapi sebuah citra dengan 3 jenis proteksi media digital.

DAFTAR PUSTAKA

[1] Guo. Jing-Ming & Chang. Chiao-Hao, (2009). *Prediction-Based Watermarking Schemes for DCT-Based Image Coding*, 2009 Fifth International Conference on Information Assurance and Security, IEEE, (P619-622).

[2] Mulyana. Teady M. S., (2013), *Penggunaan Nilai Skala Keabuan Dari Citra Watermark Sebagai Cetak Biru Dari Visible Watermarking*, Seminar Nasional Informatika 2013 Proceddings – Buku 2, UPN "Veteran" Yogyakarta, Yogyakarta , pp 23.

[3] Simarmata. J., Chandra. T., (2007), *Grafika Komputer*, Penerbit Andi, Yogyakarta.

[4] Sutoyo, T. Mulyanto, E. Suhartono, V. Nurhayati, OD. Wijanarto., (2009), *Teori Pengolahan Citra Digital*, Penerbit Andi, Yogyakarta.